6.0 SOFTWARE QUALITY ASSESSMENT

This section describes the findings of the software quality assessment (SQA) of EADSIM Version 4.01a and Version 5.00c. The SQA addressed the EADSIM defect density, programming conventions and development procedures, source code quality, computational efficiency, and supportability. These assessments were performed via (1) review of available development procedure and process documentation, (2) analysis of the post-release change request/problem report database, (3) review and desk checking of the source code, and (4) static analysis of the source code. The software quality measures of effectiveness (MOEs) used in the assessment are illustrated in Figure 6.0-1 and are defined in Table 6.0-1 along with the MOE scoring and weighting factors. A software quality score was determined for each MOE and an overall software quality rating was determined by summing the weighted scores for each MOE. Software Quality ratings were determined for both EADSIM Version 4.01a and 5.00c. The following paragraphs discuss each MOE separately and present summary results for each MOE assessment. Supporting analyses are available in separate technical reports. Finally, the SQA conclusions and implications for use are discussed in paragraph 6.6 and the summary SQA MOE scores are presented.

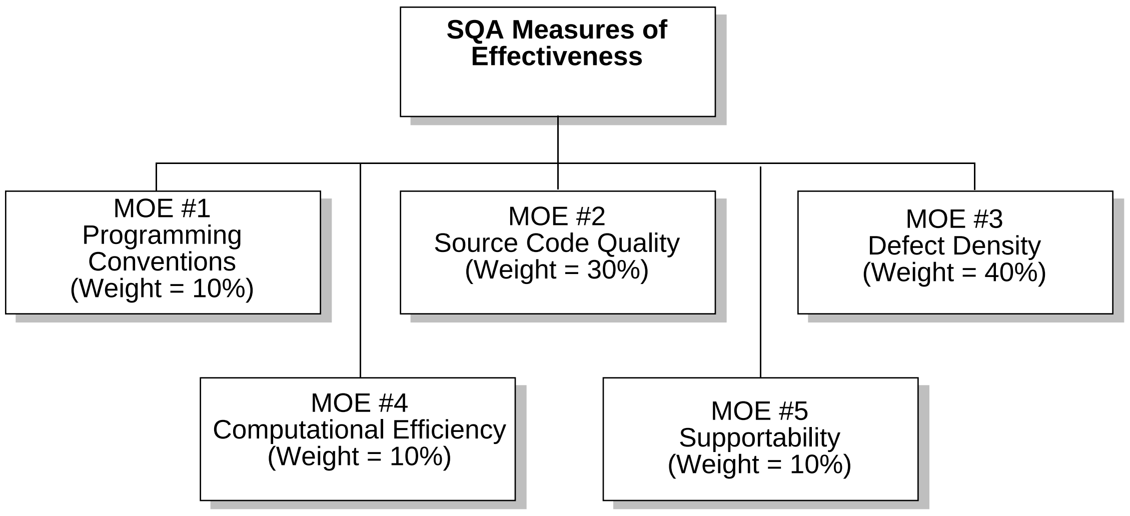
SQA Measures of Effectiveness
MOE #1 Programming Conventions (Weight = 10%)
MOE #2 Source Code Quality (Weight = 30%)
MOE #3 Defect Density (Weight = 40%)
MOE #4 Computational Efficiency (Weight = 10%)
MOE #5 Supportability (Weight = 10%)

FIGURE 6.0-1. SQA Measures of Effectiveness.

TABLE 6-1. EADSIM Software Quality MOE Definitions.

Software Quality Measures of Effectiveness (MOE)	**Poor Practice or Unacceptable (Score 0 to 5)**	**Acceptable (Score 5 to 8)**	**Excellent (Score 8 to 10)**
MOE#1 - Programming Conventions (Weight 10%)	Programming Conventions do not Exist or >2% of Code has Significant Compliance Problems	Programming Conventions Exist but are not Documented, not Consistent with Industry Standards or not Mandated as Part of Dev. Process. < 2% of Code has Significant Compliance Problems.	Programming Conventions are Documented, Consistent with Industry Standards and Mandated as Part of Dev. Process. Only Minor Compliance Problems Noted.
MOE#2 - Source Code Quality (Weight 30%) (Subfactors Weighted Equally)	1. Average Complexity (CPX) >30 2. >4% Of Functions Exceed CPX of 100 3. Ratio Comments to Code < .4 4. Desk Checks Show Major Deficiencies	1. Average Complexity (CPX) <30 2. <4% of Functions Exceed CPX of 100 3. Ratio Comments to Code > .4 4. Desk Checks Show Some Significant Deficiencies	1. Average Complexity (CPX) <10 2. <1% of Functions Examined Exceed CPX OF 100 3. Ratio Comments to Code > .8 4. Desk Checks Show Only Minor Deficiencies
MOE#3 - Defect Density (Weight 40%)	Defect Identification and Tracking Process Not Established or Post-release Defect Density >5 per 1,000 Lines of New/Changed Code	Defect Identification and Tracking Process Established and Post-release Defect Density <5 Per 1,000 Lines of New/changed Code	Defect Identification and Tracking Process Established with Formal CCB and Post-release Defect Density <2 per 1,000 Lines of New/changed Code
MOE#4 - Computational Efficiency (Weight 10%)	>5% Problem Reports Related to Computational Efficiency, Major Inefficiencies Identified by Desk Checks or Static Analysis	<5% Problem Reports Related to Computational Efficiency, Minor Inefficiencies Identified by Desk Checks or Static Analysis	No Problem Reports Related to Computational Efficiency, no Inefficiencies Identified by Desk Checks or Static Analysis
MOE#5 - Supportability (Weight 10%)	Poor or Non-existent SW Documentation, Poorly Commented Source Code, Limited Tools	Good Documentation & SW Development Plan, Adequate Tools, Well Commented Source Code	Full Requirements, Design, Code and Test Documentation, Integrated Case Tool Environment, Source Code Meets all Coding Standards

6.1 PROGRAMMING CONVENTIONS

The programming conventions (EADSIM-specific coding standards) used on EADSIM are defined in the EADSIM Software Development Plan (SDP). A review of the EADSIM C Language Coding Standards (Appendix B of the SDP) determined that the standards are generally consistent with recommended industry C coding standards (see e.g. [11]. Spot checking of the source code, such as the example illustrated in Figure 6.1-1, for conformance to the EADSIM coding standards indicated good compliance with the coding standards. The Programming Conventions MOE was rated as excellent with a score of 8.5 due to (1) only minor non-compliance problems, (2) the coding standards are reasonably consistent with industry guidelines, (3) the programming conventions/coding standards are documented and compliance with the standards is mandated and checked by the developer. The programming conventions assessment applies to both Version 4.01a and 5.00c.

The INTER.C software (part of Version 4.01 C3I) was analyzed with the following results.

1. The code was reviewed for the various loop constructs, such as FOR, DO, IF, and WHILE. There were no DO loops. Some of the FOR loops did not initialize the starting parameter (see lines 1282, 2041, 2274, and 2474)
2. There were no SWITCH statements found in the code.
3. Comments were sparse but the naming conventions for parameters and functions made the code relatively easy to read and understand.
4. Found several imbedded function calls in units conversion functions but not considered a problem.
5. Code could be made more readable if return values were named with a #define statement--return (NoSolution) is clearer than return(0).

FIGURE 6.1-1. Representative Source Code Desk Checking Results.

The assessment of the EADSIM coding standards also yielded the following recommendations for EADSIM Coding Standards/Programming Conventions improvements:

- The programming conventions could be improved by enforcing the smaller function sizes recommended by the coding standards. Some of the functions appear to be too large and would probably benefit from further decomposition.
- Some aspects of the coding standards are too vague to be useful, e.g., "C functions shall be limited in complexity to allow the desired purpose of the function to be implemented." This could be restated to specify coding standard requirements for average complexity and maximum complexity. This would improve the quality and maintainability of the code.
- Limits or guidelines for block nesting levels are not stated (e.g., don't exceed a nesting level of 7). This would help reduce the complexity of the EADSIM source code.

6.2 SOURCE CODE QUALITY

Source code quality was assessed by two related methods, (1) Complexity of the source code and (2) Ease of understanding and sufficiency of source code comments. The approach used in assessing the complexity and clarity of the source code was as follows:

a. All of the source code was evaluated using the C-DOC C source code static analyzer. A representative C-DOC summary report for the Version 4.01a Executive functions module is shown in Table 6.2-1. Results captured from C-

DOC include cyclomatic complexity (simply called complexity in this report), number of comment lines, and number of non-comment lines of code. Detailed static analysis results for Version 4.01a and 5.00c are available as separate technical reports.

b. Summary static analysis reports for EADSIM Version 4.01a and 5.00c are shown in Tables 6.2-2 and 6.2-3, respectively. The static analysis results and MOE scores are based on the static analysis reports and the limited desk checking.

c. A desk check analysis of the C3I C source code was performed, using the C-DOC measures, especially cyclomatic complexity, as indications of source code likely to be of lesser relative quality. The desk check used the check list described in "Software Engineering in the UNIX/C Environment" Appendix E by Frakes, Fox, and Nejmeh [11].

The Software Quality MOE analysis results for Version 4.01 are as follows:

- Average Complexity: Averaging the data shown in Table 6.2-2, the Version 4.01a source code has an average complexity of 16.6 across all functions. This is in the acceptable range (<30 and >10) and was scored as 7.0 out of 10. A typical complexity profile (for the flight processing functions) is shown in Figure 6.2-1.

- Percentage of High Complexity Functions: The static analysis identified 52 functions (1.66% of 3,129 total functions) with a complexity of greater than 100. The high complexity was typically the result of a larger module size and very complex coding (e.g., nested if-else blocks 7 levels deep, spanning over 4 pages of listing). Some of the complexity is inherent in the EADSIM problem but some of the complexity could be reduced by using additional functions to replace portions of the complex code. The % functions exceeding a complexity of 100 was acceptable and was assigned a score of 7.5.

- Ratio of Comment Lines to Non-Comment Lines of Code: The comments to code ratio factor for EADSIM Version 4.01a is 47%. This is a marginally acceptable level of comments and was scored as 5.5.

- Severity of Problems Identified by Desk Checking the Source Code: The desk checking (discussed above) revealed several minor coding deficiencies and was scored as 8.5.

EADSIM Version 5.00c received almost identical scores to Version 4.01a in the area of software quality. The average complexity increased slightly to 17.3 while the percentage of high complexity functions decreased slightly. The percentage of source code comments remained at 47%. The limited desk checking of source code still indicated some minor coding deficiencies but none were considered significant. Some very slight trends can be determined from the comparison of Version 4.01 and 5.00, including a trend for increasing source code complexity and an unchanging level of source code commenting.

TABLE 6.2-1. EADSIM Version 4.01a Example Static Analysis Report (EADSIM Executive)

FUNCTION	CPX	STMTS	LOC	CMNTS	LINES	MODULE
<null>	0	131	119	196	454	C3ISIM.C
BruteSearch	5	21	30	17	50	RUNSIM.C
CatchRunTimer	1	6	8	17	27	RUNSIM.C
CheckForDuplicates	3	12	18	11	37	EXECUTIL.C
CheckProcessFiles	12	55	71	35	122	RUNC3I.C
CleanLastRun	41	122	204	3	217	RUNC3I.C
ClearEntry	4	23	27	24	62	FIND.C
DefaultPref	2	38	41	23	77	RUNOPTIO.C
DisplayCopyRightWindow	4	79	99	34	147	READIMG.C
DisplayFile	8	40	60	36	115	REQUESTE.C
DisplayRGBLogo	1	23	18	32	155	READIMG.C
ExecFunctions	1	169	183	26	228	EFUNCT1.C
FileDisplay	4	22	28	25	67	FILESCAN.C
FileScan	4	12	18	22	47	FILESCAN.C
FileSelect	21	98	140	51	219	REQUESTE.C
FileSelect2	22	102	145	54	232	REQUESTE.C
FindEntry	12	52	63	31	116	FIND.C
FindProcessNumber	7	22	38	22	65	RUNC3I.C
GetPath	2	11	11	15	31	EXECUTIL.C
GetScenarioName	10	46	66	33	119	RUNOPTIO.C
Help	3	9	10	2	14	EXECUTIV.C
InitRunSim	2	11	13	21	38	RUNSIM.C
KillAllRunTime	5	10	12	3	15	RUNC3I.C
KillProcess	3	29	35	3	44	RUNC3I.C
MenuItemDefault	1	4	7	2	10	EXECUTIV.C
MenuPick	3	10	14	7	24	EXECUTIV.C
MessageLine	2	10	12	17	34	EXECUTIV.C
MultiStringSelect	13	62	72	44	150	REQUESTE.C
MultiStringSelectAll	15	66	79	52	165	REQUESTE.C
OpenRunOptions	4	23	35	23	71	RUNOPTIO.C
PStrCmp	1	4	6	10	18	REQUESTE.C
ProcessError	8	24	32	8	46	EXECUTIV.C
QuitRunOption	1	8	10	17	31	RUNOPTIO.C
ReplaceBlanks	3	6	10	13	28	EXECUTIL.C
RunC3ISIMExecute	45	212	289	33	444	RUNC3I.C
SaveRunOptions	1	25	27	18	51	RUNSIM.C
SendRunFile	3	29	36	37	82	RUNC3I.C
SendTBRunFile	2	49	69	30	102	RUNC3I.C
SetDefaultRunOptions	1	10	12	18	33	RUNOPTIO.C
SetRunOptions	6	74	93	20	123	RUNOPTIO.C
SimpleRequest	33	152	181	35	261	REQUESTE.C
StartRunTime	20	124	159	29	231	RUNSIM.C
StringSelect	13	80	107	52	203	REQUESTE.C
StringSelect2	11	69	88	48	176	REQUESTE.C
TokenDown	3	10	14	6	22	EXECUTIV.C
VerifyRun	11	28	57	18	77	RUNSIM.C
executive	5	15	21	28	58	EXECUTIV.C
main	1	15	19	28	60	C3ISIM.C
SUMMARY						
TOTAL	383	2252	2906	1329	5198	
AVERAGE	8.0	47	61	28	108	
MAXIMUM	45	212	289	196	454	

TABLE 6.2-2. EADSIM Version 4.01a Static Analysis Results.

Build Name	Total LOC	Total Comments	Average Complexity	Maximum Complexity	Maximum LOC
EXECUTIVE	2906	1329	8.0	45	289
BRAWLER	2216	421	3.7	50	298
CONFEDERATION	3466	1891	8.6	52	292
C3I	54407	34453	16.2	285	2007
FLIGHT PROCESSING	18889	9431	25.1	256	1536
PLAYBACK	1631	749	12.7	89	488
POST-PROCESSING	21778	5236	29.2	209	807
PROPAGATION	2761	1715	14.6	98	463
SENSORS	10783	6792	18.5	128	911
SCENARIO GENERATION	103003	43493	10.8	250	3197
SCENARIO FORMATTING	13488	1594	42.2	585	4119
TERRAIN INT. ROUGH EARTH MODEL	2959	1880	10.2	81	559
TRAJECTORY TOOLS	2673	1582	27.9	150	769
UTILITIES	36619	16945	15.1	271	2058
WINDOWS MANAGER	8026	4349	6.3	101	633
TOTALS	285,605	131,860			

TABLE 6.2-3. EADSIM Version 5.00c Static Analysis Results.

Build Name	Total LOC	Total Comments	Average Complexity	Maximum Complexity	Maximum LOC
EXECUTIVE	2921	1347	8.0	45	294
BRAWLER	2384	778	3.7	50	316
CONFEDERATION	3733	2095	8.3	52	306
C3I	74849	44545	17.6	390	2343
FLIGHT PROCESSING	22140	11094	20.8	350	2074
PLAYBACK	2007	907	15.3	134	675
POST-PROCESSING	22051	5359	29.8	214	807
PROPAGATION	2447	1332	15.2	98	466
SENSORS	15583	9738	17.6	114	848
SCENARIO GENERATION	121218	50946	10.8	347	3127
SCENARIO FORMATTING	19394	2170	51.4	674	4643
TERRAIN INT. ROUGH EARTH MODEL	2964	1932	10.2	81	559
TRAJECTORY TOOLS	2679	1583	27.9	150	769
UTILITIES	44556	20426	16.3	384	2858
WINDOWS MANAGER	8094	4380	6.4	101	633
TOTALS	347,020	158,632			

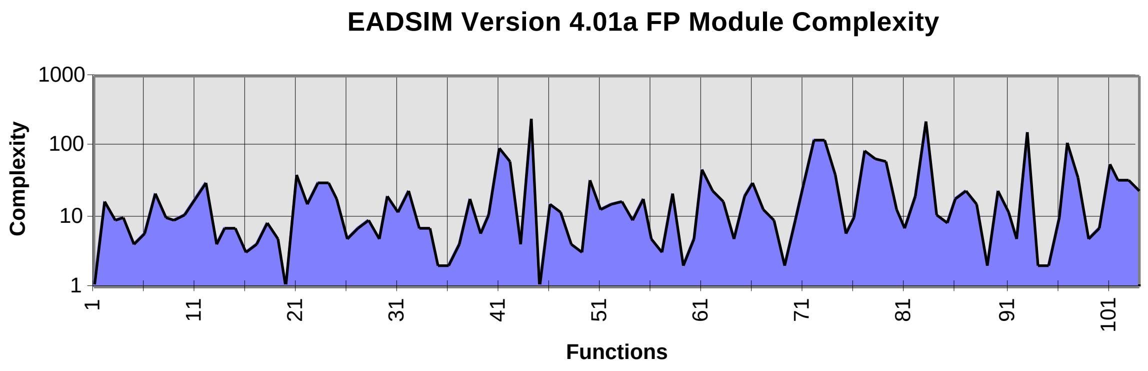

FIGURE 6.2-1. Complexity Profile Example (Version 4.01 Flight Processing).

6.3 DEFECT DENSITY

Defect density is defined as the number of errors or defects identified in a software release divided by the number of new and modified source lines of code in the release. Defects identified and corrected prior to the version release to the user community are not included in the defect count. The approach taken to determine defect density for EADSIM was to review the change requests (CRs) in the EADSIM CR database to determine which CRs were enhancements and which were defects. Change requests are the mechanism the developer (TBE) and the SSDC Testbed Product Office use to track post-release problems and requested enhancements. The EADSIM change request/problem report process has been in place for a number of years and has proven to be an effective process for capturing and tracking identified problems and requested enhancements.

Table 6.3-1 lists the CRs identified as defects in EADSIM Version 4.01. Figure 6.3-1 shows the defect profile for Version 4.01. Defect density could not be accurately computed for Version 4.01 since a reliable estimate of the new and changed source code included in Version 4.01 was not available. A rough estimate of Version 4.01 defect density was made by assuming that Version 4.01 included 18,000 lines of new and changed source code. This resulted in an estimated defect density of slightly less than 2 defects per 1,000 lines of new/changed source code. The Version 4.01 defects per 1,000 lines of total source code was less than .13 or about 1 defect per 8,000 lines of code. Given that high quality commercial software typically strives for less than one defect per thousand lines of new/changed source code, the EADSIM defect density estimate indicates that the Version 4.01 release achieved a level of quality on par with good to excellent quality commercial software. The Defect Density MOE for Version 4.01 was scored a 9.0 (excellent).

TABLE 6.3-1. EADSIM Version 4.01 Post-Release Defect Listing.

Date Reported	Defect #	Status	Functional Area	SCR No.
12/7/94	1	Incorporated in 4.02	MOEs	10181
12/7/94	2	Incorporated in 4.02	MOEs-Detection	10182

TABLE 6.3-1. EADSIM Version 4.01 Post-Release Defect Listing.

Date Reported	Defect #	Status	Functional Area	SCR No.
12/15/94	3	Incorporated in 4.02	Directed Energy Weapons	10184
1/12/95	4	Incorporated in 4.02	User Interface	10185
1/15/95	5	Incorporated in 4.02	User Interface	10187
1/15/95	6	Incorporated in 5.00	EW-Jammers	10188
1/15/95	7	Incorporated in 4.02	User Interface	10192
1/15/95	8	Incorporated in 4.02	Electronic Warfare (EW)	10193
1/15/95	9	Incorporated in 4.02	General	10194
1/20/95	10	Incorporated in 4.02	MOEs-Comm	10195
1/20/95	11	Incorporated in 4.02	User Interface	10196
1/23/95	12	Incorporated in 4.02	Attack Ops	10197
1/24/95	13	Incorporated in 4.02	User Interface	10198
1/25/95	14	Incorporated in 4.02	User Interface	10199
1/25/95	15	Incorporated in 4.02	User Interface	10200
1/25/95	16	Incorporated in 4.02	User Interface	10201
1/26/95	17	Incorporated in 4.02	User Interface	10202
2/28/95	18	Incorporated in 4.02	User Interface	10222
3/6/95	19	Incorporated in 4.02	User Interface	10223
3/6/95	20	Incorporated in 4.02	User Interface	10224
3/8/95	21	Approved	Air	10235
3/14/95	22	Undecided	Air	10238
3/14/95	23	Incorporated in 5.00	User Interface	10243
3/21/95	24	Incorporated in 4.02	Aircraft Flight	10246
4/14/95	25	Incorporated in 5.00	DIS/ALSP	10267
4/14/95	26	Incorporated in 4.02	User Interface	10269
4/14/95	27	Incorporated in 4.02	User Interface	10270
5/11/95	28	Incorporated in 5.00	User Interface	10293
6/18/95	29	Incorporated in 4.02	User Interface	10297
11/27/95	30	Incorporated in 5.00	EW-Jammers	10438
1/9/96	31	Undecided	General	10454
1/9/96	32	Incorporated in 4.02	User Interface	10456
1/9/96	33	Undecided	DIS/ALSP	10460
4/18/96	34	Undecided	10519	

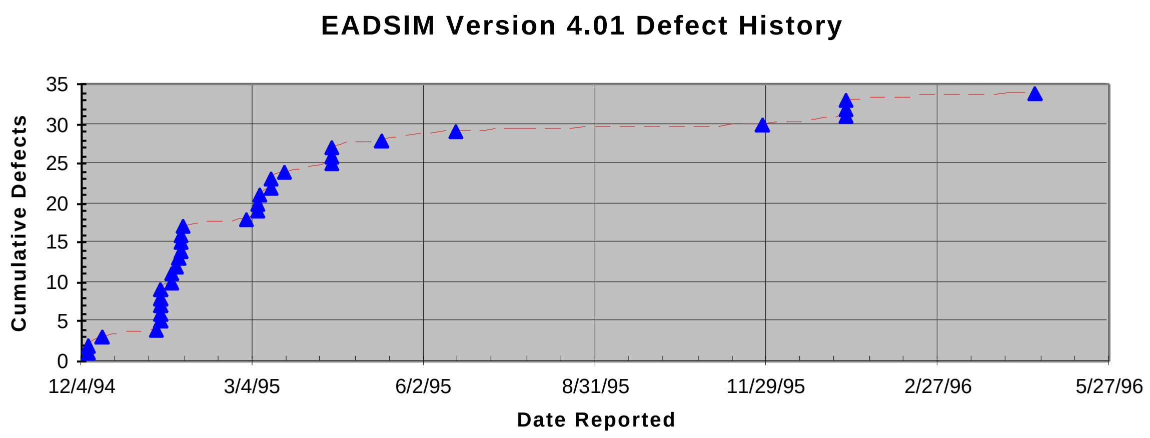

FIGURE 6.3-1. EADSIM Version 4.01 Defect Profile.

Version 5.00's profile is similar to the Version 4.01 in the number of defects, however the 5.00 defect profile is more linear than Version 4.01. The total defects per 1,000 lines of new code for Version 5.00 was .59 (36 defects divided by 61,415 new lines of code). This indicates an excellent level of quality and the MOE was scored as 9.0. The defects per 1,000 lines of total source code was about .10 or which is equivalent to 1 defect per 10,000 lines of code. Both the new code defect density and the total code defect density show an improvement over Version 4.01 but not enough improvement to justify a higher MOE score. A list of the Version 5.00 defects is shown in Table 6.3-2. These are plotted in Figure 6.3-2.

TABLE 6.3-2. EADSIM Version 5.00 Defect List.

Date Reported	V5.00 Defect	Status	Functional Area	SCR No.
1/11/96	1	Incorporated in 5.00a		10464
1/12/96	2	Incorporated in 5.00a		10465
1/12/96	3	Undecided	User Interface	10466
1/15/96	4	Incorporated in 5.00a		10467
1/16/96	5	Incorporated in 5.00a		10468
1/17/96	6	Undecided	User Interface	10469
2/6/96	7	Incorporated in 5.00b		10473
2/8/96	8	Incorporated in 5.00b		10474
2/19/96	9	Incorporated in 5.00b		10475
2/19/96	10	Incorporated in 5.00c		10476
2/23/96	11	Incorporated in 5.00b		10479
2/26/96	12	Incorporated in 5.00b		10480
2/28/96	13	Incorporated in 5.00b		10481
2/28/96	14	Incorporated in 5.00b		10482
3/5/96	15	Undecided		10484

TABLE 6.3-2. EADSIM Version 5.00 Defect List.

Date Reported	V5.00 Defect	Status	Functional Area	SCR No.
3/6/96	16	Incorporated in 5.00b		10486
3/11/96	17	Incorporated in 5.00c		10487
3/12/96	18	Incorporated in 5.00c		10488
3/13/96	19	Undecided		10490
3/14/96	20	Incorporated in 5.00c		10491
3/14/96	21	Undecided		10492
3/19/96	22	Undecided		10493
3/20/96	23	Undecided		10495
3/20/96	24	Incorporated in 5.00c		10496
3/21/96	25	Undecided		10498
3/26/96	26	Incorporated in 5.00c		10499
3/26/96	27	Undecided		10500
3/26/96	28	Incorporated in 5.00c		10509
3/26/96	29	Incorporated in 5.00c		10510
4/5/96	30	Undecided		10512
4/5/96	31	Undecided		10513
4/5/96	32	Undecided		10514
4/5/96	33	Undecided		10515
4/5/96	34	Undecided		10516
4/16/96	35	Undecided		10517
4/16/96	36	Undecided		10518

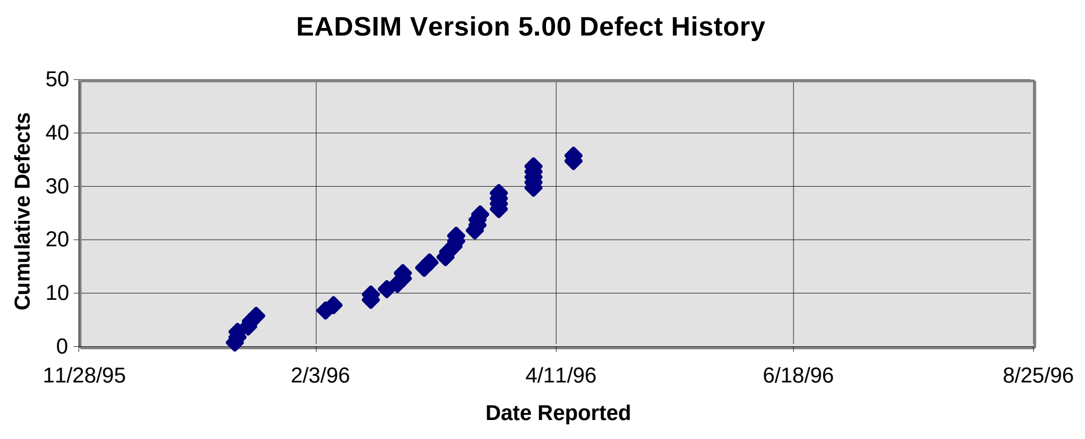

FIGURE 6.3-2. EADSIM Version 5.00c Defect Profile.

A comparison of the Version 4.01 and 5.00 defect profiles was performed. The total number of post-release defects were very similar though the profiles are somewhat different. The defect density for Versions 4.01 and 5.00 are shown in Figure 6.3-3, plotted along with the interim Version 4.02 profile. Version 4.02 was not an official EADSIM release and is only shown here for completeness. Version 4.02 supported a number of quick response TMD COEA requirements, hence there was a higher level of post-release defects.

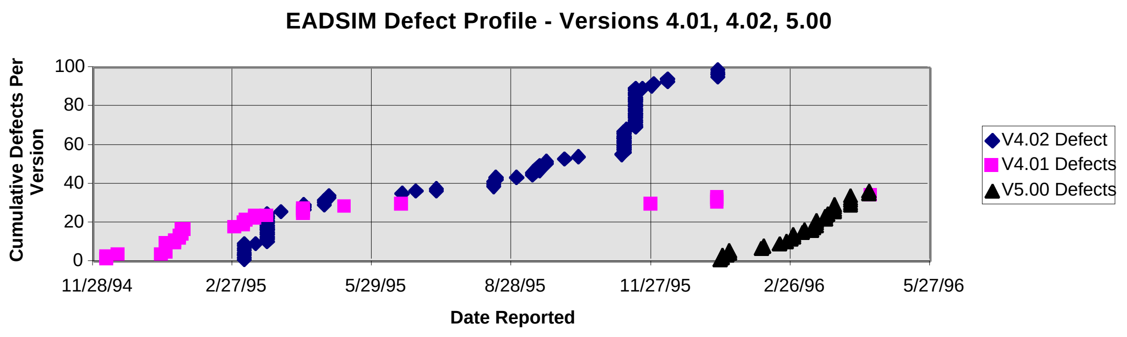

FIGURE 6.3-3. Defect Profile for Versions 4.01, 4.02 and 5.00.

The EADSIM Versions 4.01 and 5.00 Defect Density MOE scores of 9.0 indicate an excellent level of software quality, especially given the size and complexity of EADSIM. The EADSIM post-release defect density could be further reduced by improvements such as:

a. improved design and source code quality (more comments, better design documentation, more use of automated tools),

b. reduced source code complexity (source code complexity on EADSIM is acceptable but high),

c. increased pre-release developer and IV&V testing,

d. increased Beta-testing by the user community (EADSIM already makes extensive use of user Beta-testing),

e. increased the level and quality of software reuse (some reused code such as the TIREM code does not appear to be well documented), and

f. link the EADSIM Change Request system to software quality-related metrics to allow correlation of problems and defects with code complexity, module size, number of comments, and other indicators and metrics. Specific source code modules and functions could then be targeted for improvement or special attention during testing.

6.4 COMPUTATIONAL EFFICIENCY

Computational efficiency includes the efficiency of the EADSIM pre-processing, run-time, and post-processing functions. Computational efficiency can be assessed by measuring the execution time for key EADSIM functions and processes or by measuring overall time to execute applicable scenarios. This level of assessment was not feasible, so an approach was used that relied on (1) an analysis of the EADSIM source code for possible computational inefficiencies and (2) assessment of user problem reports to determine if users have reported computational efficiency problems or concerns. Approach (1) was applied to the latest EADSIM source code version and approach (2) was applied to versions 4.01 through 5.00.

Version 5.00c source code was reviewed for algorithm efficiency and for efficient control of memory allocation and deallocation. In both cases, random checking of the source code was used due to the large size of source code to be analyzed. No specific algorithm or memory management inefficiencies were identified.

The EADSIM change request database was reviewed for references to slow run-time, memory allocation problems, and other inefficiencies for Version 4.01 and later. No user problems were reported in these areas.

Based on the above findings, the computational efficiency MOE was rated at 9.5.

6.5 SUPPORTABILITY

Software supportability includes all software factors that affect the maintenance or supportability of the software product. These factors include software support planning, product documentation, product complexity, software support environment, clarity of the source code, and availability of personnel with product design knowledge. This MOE was assessed by reviewing the EADSIM Software Development Plan and by analyzing the metrics data collected for the software quality and defect density MOEs. The EADSIM supportability assessment results were as follows:

- EADSIM has a good software development plan and a workable software support process, including an effective hot-line and a good problem report/change request tracking system.
- Product documentation is good but limited to user and methodology documentation. The design documentation needs improvement but is adequate.
- EADSIM complexity is high. Some of the complexity could be reduced, some is inherent in the EADSIM problem domain.
- The EADSIM software development and support environment (both Sun and Silicon Graphics) is a proven environment that is readily available. The recommended area of improvement would be the use of additional software support tools (e.g., static and dynamic code analyzers).
- The EADSIM source code is reasonably clear and generally adheres to the coding standards. However, some functions are very complex and the level of commenting is generally adequate but not excellent.
- There are a number of EADSIM-knowledgeable personnel to support the maintenance and development efforts. A reasonably large base of trained, knowledgeable personnel are resident in the developer (TBE) and the user community.

The above findings produced an acceptable MOE rating of 7.5.

6.6 IMPLICATIONS FOR MODEL USE

The SQA MOE results for EADSIM Version 4.01a and 5.00c are summarized in Table 6.6-1. The overall software quality of EADSIM Version 4.01a and 5.00c was assessed as excellent with a score of 8.34 out of a possible 10. The relatively low number of post-release defects, the proven software development process, good user and methodology documentation and generally understandable source code are EADSIM's software quality strong points. EADSIM's primary software quality weak points are a relatively high source code complexity, limited design documentation, and a marginally acceptable level of source code commenting.

TABLE 6.6-1. EADSIM Version 4.01a SQA MOE Results.

Software Quality Measures Of Effectiveness (MOE)	**Poor Practice or Unacceptable (Score 0 to 5)**	**Acceptable (Score 5 to 8)**	**Excellent (Score 8 to 10)**
MOE#1 - Programming Conventions (Weight 10%)			Score: 8.5 Coding Standards Documented in the SDP, Generally Consistent With Industry Standards, Source Code Generally Follows the Coding Standards.
MOE#2 - Source Code Quality (Weight 30%) (Subfactors Weighted Equally)		Score: 7.13 1. Average Complexity (CPX) 16.6 for Version 4.01 and 17.3 for Version 5.00. [Score 7.0] 2. 1.7% of Functions (52 out of 3,129 for Version 4.01) Exceed CPX OF 100 [Score 7.5] 3. Ratio Comments to Code = 47% for Versions 4.01 and 5.00 [Score 5.5]	4. Limited Desk Checks For Versions 4.01 and 5.00 Show Only Minor Deficiencies [Score 8.5]
MOE#3 - Defect Density (Weight 40%)			Score: 9.0 Defect Identification and Tracking Process Established with Formal CCB and Version 5.00 Post-release Defect Density of .59 Defects per 1,000 Lines of New Code

TABLE 6.6-1. EADSIM Version 4.01a SQA MOE Results.

Software Quality Measures Of Effectiveness (MOE)	**Poor Practice or Unacceptable (Score 0 to 5)**	**Acceptable (Score 5 to 8)**	**Excellent (Score 8 to 10)**
MOE#4 - Computational Efficiency (Weight 10%)			Score: 9.5 No Problem Reports Related to Computational Efficiency, No Inefficiencies Indicated by Desk Checks or Static Analysis Except for High Cyclomatic Complexity
MOE#5 - Supportability (Weight 10%)		Score: 7.5 Good Documentation & SW Development Plan, Adequate Tools, Source Code Commenting Marginally Acceptable.	

The software quality-related implications for use are:

- EADSIM defect density is relatively low, hence prospective users can reasonably expect software releases with few defects. The defect density for Version 5.00c is less than one defect per thousand lines of added source code, which indicates excellent software quality.
- EADSIM is supported by an effective change request/problem report tracking and reporting system, which feeds into a formal CCB driven CM process. This gives users visibility into the identified problems and requested enhancements as well as a mechanism to report problems to the development organization.
- The EADSIM source code, though complex, is adequately documented and understandable. However, a significant user investment in training and EADSIM-knowledgeable personnel is required if the user wishes to fully understand and/or modify the EADSIM source code. This is due to (1) the large software size (over 340,000 non-comment source lines of code), (2) the relatively high source code complexity, (3) an acceptable but less than excellent ratio of source code comments to executable source code, and (4) lack of user-accessible detailed design documentation. This learning curve is somewhat reduced by the good documentation at the user and methodology level and the knowledgeable and responsive TBE development team.